# 综述

姓名：蒋慧 学号：20158866 专业：计算机科学与技术201506电话：15520253763 邮箱：[1324789616@qq.com](mailto:1324789616@qq.com)

简介：我大一上学期跟着学校的课程学习了c语言基础，

然后大二下期开始接触网页制作 ，然后通过在网上看视频自学了html css和javascript 并跟着做了一些demo（我提交的网页demo是以前制作的）大二上学期继续网页，最近开始学习Java语言 （java代码是面试后敲的）。

认识：

我觉得java是一门强大的语言，他比C++更加智能化而且比较成熟，还有它的跨平台性就完胜.NET，java的应用也很广，许多大型网站都是用java编写的，还有手机App开发 ，安卓开发，数据库等，现在狂热的迷上了java。希望可以循着java这条路上好好走下去。

规划 ：

我现在主要是想往java方面发展，之后和web方面结合，想加入工作室，进入一个圈子，然后多一些实战，而不是仅仅停留与理论，还有就是想跟着大神们学习学习。

网页demo简介：

这是一个主要用css制作的页面滑动效果，body部分设置overflow-hidden底部的选项是通过fixed定位input单选按钮和a锚点标签制作，鼠标放上去变色，并且出现三角形（通过伪类和边框制作，让边框透明在设置bottom颜色）主体页面滑动是利用外面的div 改变translateY使页面滑动，用transition控制时间和移动速度，页面上的三角则使用div制作，用绝对定位和margin使其居中，再用transform让其旋转，上移让页面仅剩下一个三角形，用data-icon设置div的自定义属性，然后用属性选择器和伪类制作三角形中的图形（用@font-face引入特殊字体使其变成相应的图标）h2和p标签都是用绝对定位加上margin line-height使其居中用text-shadow设置文字阴影，用@Keyframes设置页面出来时h2和p跳动的动画效果，并用transition引入动画 设置时间等@media screen设置自适应的屏幕大小。

Java 学生-选课表 多对多关系简介：

首先创建一个抽象类，再建一个学生和课程的类继承自抽象类，一个用private封装学生和课程的属性，一个学生有多门课程，一门课程有多个学生，并且不同学生的课程数量不相同，不同课程的学生个数不相同，所以需要建立一个链表类，对学生的数量和课程数量进行保存。

抽象类：

主要是实现实例化对象的向上转型，以适应链表的参数，抽象类包括get（）， set（），compare（），以及search（）抽象方法必须在子类中进行覆写。

Link类：

主要用递归和内部类（内部类实例可以方便的访问外部类）实现，Link类对象用来保存节点数据，以及对节点的操作，包括增加节点addNode(),查询节点containNode(),设置索引节点的数据setIndex(),得到索引节点数据getIndex(),删除节点removeNode()，将节点数据存放在数组里ArrayNode()，外部类有一个根节点root，count表示链表长度，每次增加节点count++； retArray[]存放链表转换后的数组，foot表示链表的索引，便于链表转换为数组和用索引查询时使用，增加节点add(),返回链表长度size(),判断链表是否为空isEmpty(),查询contains()，得到索引处数据get()，设置索引处数据set()，删除remove()链表变数组toArray()

工厂类：便于客户端的使用，用户不用知道具体要创建什么类的对象，只要使用工厂类的static方法 并传入参数就可以了。

学生类：

有学号 姓名 性别 年龄属性，还要创建一个Link对象用来保存学生的课程，还要有获得属性的方法，得到链表以及设置链表的方法，以及进行查询的从父类继承的方法search（）的覆写以及重载。

课程类：

有课程编号 课程名，教师姓名，教师性别属性，和一个Link类实例化对象，用来存放该课程的学生，获得属性的方法以及对父类方法search（）的覆写与重载。

------------------------------------------------------------------------------------在主方法中：

System.out.println("对象名's members:"+ 对象名.get().size());输出相应链表的长度

System.out.println("C1's Students:");

for(int i=0;i<c1.get().toArray().length;i++){

System.out.println(c1.get().toArray()[i].getImfo());

}

将链表变为数组循环输出学生的课程信息或者课程的学生信息

try{

try{

if(args[0].length()==5){

Object []obj1=c1.search(Integer.parseInt(args[0])).toArray();

for(int i=0;i<obj1.length;i++){

Students pp=(Students)obj1[i];

System.out.println("c1:"+pp.getImfo());

}

Object []obj2=c2.search(Integer.parseInt(args[0])).toArray();

for(int i=0;i<obj2.length;i++){

Students pp=(Students)obj2[i];

System.out.println("c2:"+pp.getImfo());

}

Object []obj3=c3.search(Integer.parseInt(args[0])).toArray();

for(int i=0;i<obj3.length;i++){

Students pp=(Students)obj3[i];

System.out.println("c3:"+pp.getImfo());

}

Object []obj4=c4.search(Integer.parseInt(args[0])).toArray();

for(int i=0;i<obj4.length;i++){

Students pp=(Students)obj4[i];

System.out.println("c4:"+pp.getImfo());}

}

}

catch(NullPointerException e)

{System.out.println("user enter error\_\_\_\_\_5");}

finally{

try{

if(args[0].length()==4||(args[0].length()==6)){

Object []obj2=c2.search(args[0]).toArray();

for(int i=0;i<obj2.length;i++){

Students pp=(Students)obj2[i];

System.out.println("c2:"+pp.getImfo());}

Object []obj1=c1.search(args[0]).toArray();

for(int i=0;i<obj1.length;i++){

Students pp=(Students)obj1[i];

System.out.println("c1:"+pp.getImfo());}

Object []obj3=c3.search(args[0]).toArray();

for(int i=0;i<obj3.length;i++){

Students pp=(Students)obj3[i];

System.out.println("c3:"+pp.getImfo());}

Object []obj4=c4.search(args[0]).toArray();

for(int i=0;i<obj4.length;i++){

Students pp=(Students)obj4[i];

System.out.println("c4:"+pp.getImfo());}

}

}

catch(NullPointerException e)

{System.out.println("user enter error\_\_\_\_4 6");}

finally{

try{

if(args[0].length()==3){

Object []obj1=x1.search(Integer.parseInt(args[0])).toArray();

for(int i=0;i<obj1.length;i++){

Courses pp=(Courses)obj1[i];

System.out.println("x1:"+pp.getImfo());

}

Object []obj2=x2.search(Integer.parseInt(args[0])).toArray();

for(int i=0;i<obj2.length;i++){

Courses pp=(Courses)obj2[i];

System.out.println("x2:"+pp.getImfo());

}

Object []obj3=x3.search(Integer.parseInt(args[0])).toArray();

for(int i=0;i<obj3.length;i++){

Courses pp=(Courses)obj3[i];

System.out.println("x3:"+pp.getImfo());

}

Object []obj4=x4.search(Integer.parseInt(args[0])).toArray();

for(int i=0;i<obj4.length;i++){

Courses pp=(Courses)obj4[i];

System.out.println("x4:"+pp.getImfo());

}

Object []obj5=x5.search(Integer.parseInt(args[0])).toArray();

for(int i=0;i<obj5.length;i++){

Courses pp=(Courses)obj5[i];

System.out.println("x5:"+pp.getImfo());

}

}

}

catch(NullPointerException e)

{System.out.println("user enter error \_\_\_3");}

}

}

}

catch(ArrayIndexOutOfBoundsException e)

{System.out.println("user not enter");}

catch(NumberFormatException e)

{System.out.println("user enter error");}

catch(Exception e){

e.printStackTrace();

}以上代码为用户输入的模糊查询以及异常抛出的处理，第一层try{….}catch(){…}catch(){…}catch(){…}为用户不输入或者输入格式不正确的处理，第二层包含有三个try…catch…finally语句用来处理查询后没有返回结果即NullPointerException异常

模糊查询仅支持查询学号 课程号或性别，而且是使用用户输入的第一个字符串args【0】进行搜索。模糊查询通过判断args[0].length()来判断是查询学生的学号（args[0].length()==5）还是课程的编号（args[0].length()==3）还是性别（args[0].length()==4||args[0].length()==6）。